Akenti Applications

Mary R. Thompson, Gary Hoo, Keith Jackson
Srilekha S. Mudumbai, Abdeliah Essiari,
William Johnston

http://www-itg.lbl.gov/Akenti/
Imaging and Distributed Collaboration Group
Lawrence Berkeley National Laboratory, Berkeley, CA 94720

Introduction

Akenti is an access control system designed to address the issues raised in permitting access to distributed resources that are controlled by multiple remote stakeholders. Akenti enables stakeholders securely to create and to distribute instructions authorizing access to their resources. Akenti makes access control decisions based on a set of digitally signed documents that represent the authorization instructions. Public-key infrastructure and secure message protocols provide confidentiality, message integrity, and user identity authentication, during and after the access decision process.

Akenti has been integrated into several different resource-control servers. The DOE 2000 Diesel Combustion Collaboratory[1] is using a version of the Apache/SSLeay web server with Akenti to provide strong access control to the experimental results of some of its members. The DOE Materials MicroCharacterization Collaboratory[2] is planning to use the Orbix SSL-enabled ORB with Akenti to control access to a remote microscope. Akenti also provides access control for a demonstration camera control server that is implemented using the CIF[3] protocols. Currently under design are secure compute and network bandwidth servers that will be based on the Globus resource management specification with Akenti providing the access control. Finally, a mobile agent system will incorporate Akenti for authenticating and enforcing the access rights of agents as they migrate to different hosts.

Secure Web server

The standard Web authorization mechanism relies on usernames and passwords. Because the passwords are transmitted unencrypted, this mechanism is inherently insecure on an open network. We have replaced the SSL-enabled Apache[4] Web server's default authorization module with one that uses Akenti. This combination, based on the SSL[5] messaging protocol and Akenti's certificate-based access control, provides strong authentication and authorization.

To view a document or to run a script, a browser uses the HTTPS protocol to submit the user's distinguished name (DN) along with the name of the resource (document or script). Apache passes this information to Akenti, which determines whether the user meets the policy requirements for accessing the resource. If so, the user is given access to the resource. Otherwise the user is advised that access was forbidden. In either case, Akenti is invisible: the user is never aware that Akenti was invoked.

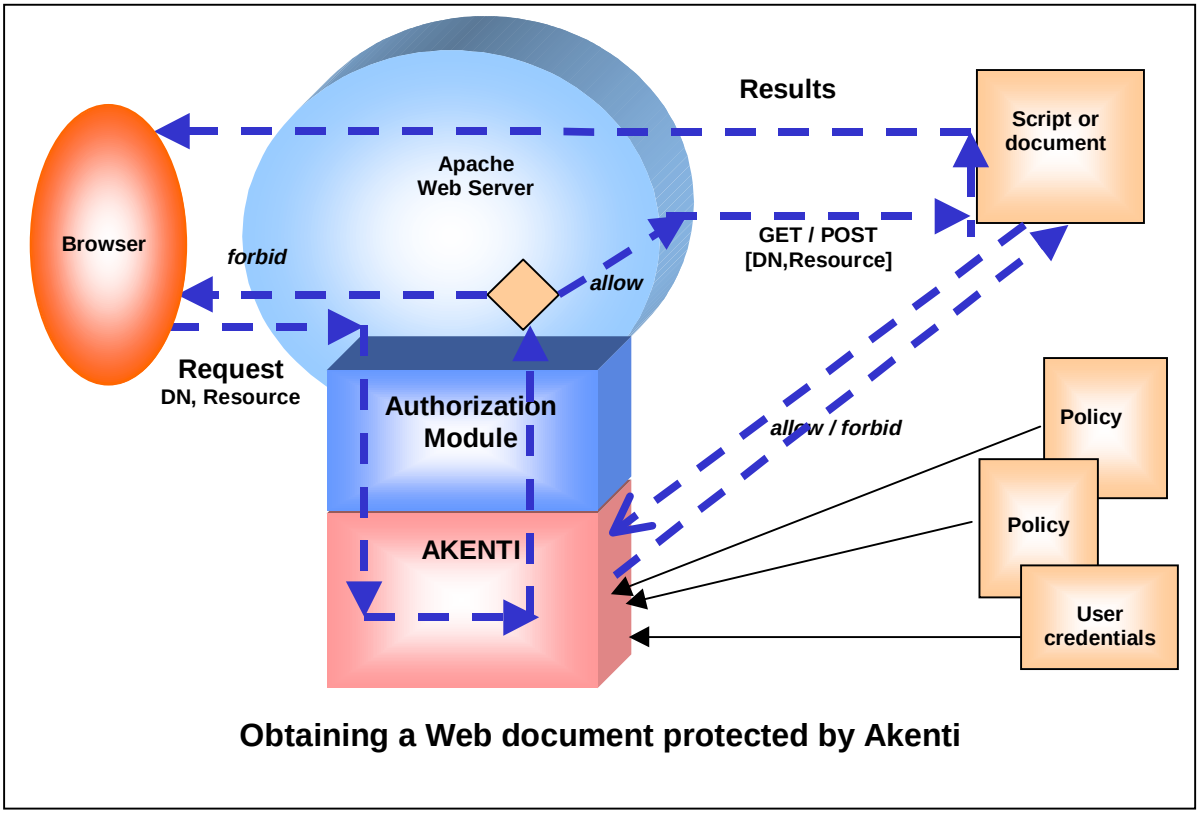
If a script protects access to other resources, the script will invoke Akenti with the resource and the user's DN to see what actions are allowed on this resource. Possible actions include modifying a document, executing a program, or accessing a database. The script will allow or deny access based on the actions that Akenti returns for the user. As above, Akenti is invisible to the end user.

Secure camera controller

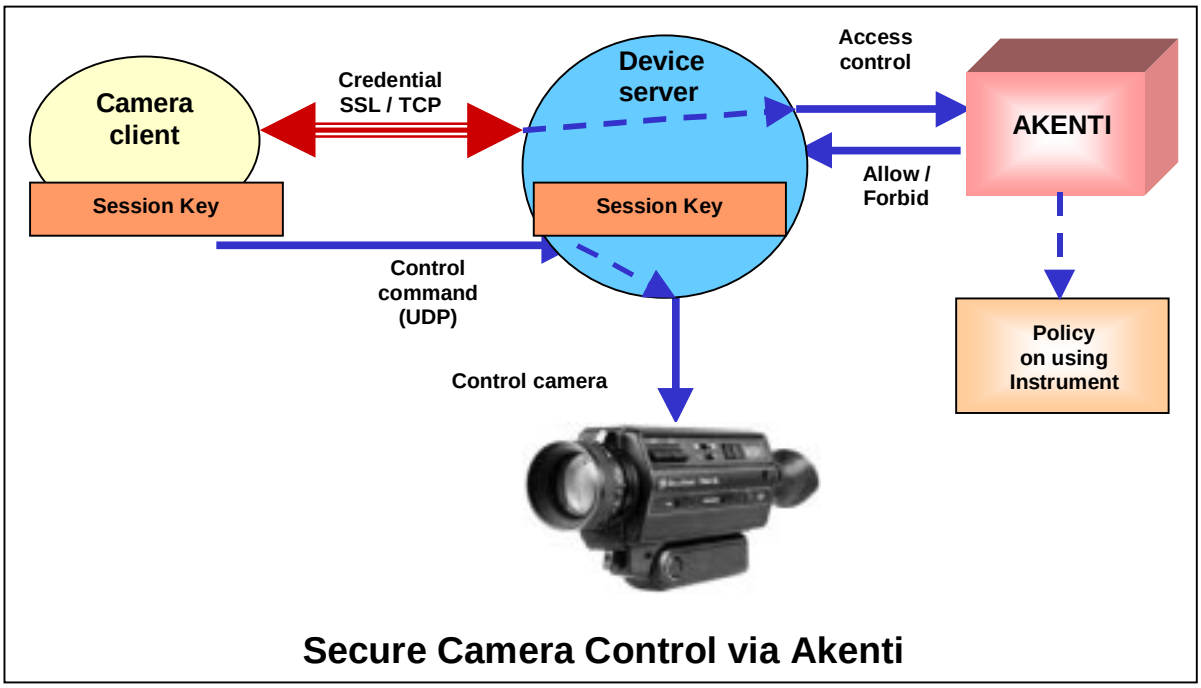
Akenti has been integrated with a camera remote-control system, enabling collaborating researchers to manipulate a remote camera from their workstations. What is of special interest is that the system uses a connectionless communication protocol whose contents can be authenticated as a result of Akenti's use. SSL is implemented on top of TCP; what was needed was a method of transmitting secure information

over UDP, since the application uses UDP to send the camera control messages.

The camera client (remote user) and device server (devserv)[6] authenticate one another and establish a shared secret via a standard TCP-based SSL handshake. The shared secret subsequently is used to hash the UDP datagrams which the camera client sends to devserv to control the camera. The hash provides a level of



Obtaining a Web document protected by Akenti



Secure Camera Control via Akenti

security because only a party in possession of the shared secret can replicate the hash and thereby confirm the message sender. If needed, the datagrams may even be encrypted, although the current system does not do so.

CORBA[7] server with Akenti access control

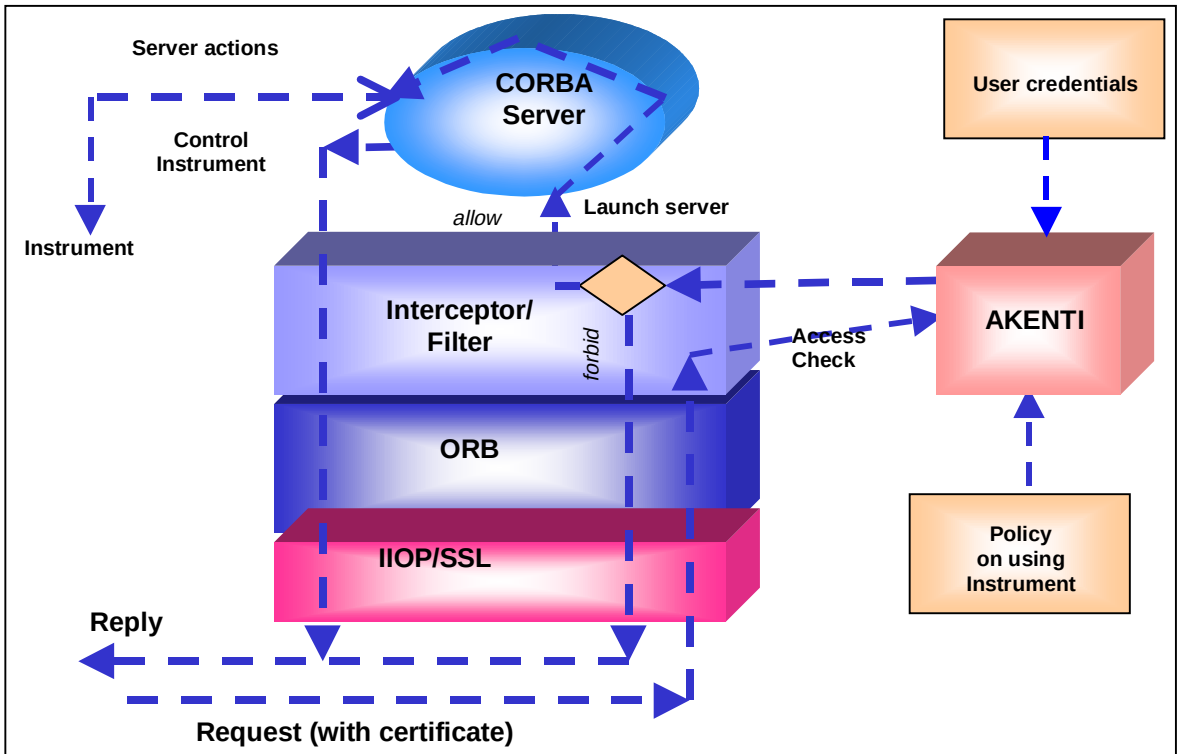
Akenti has also been used to control access to CORBA-brokered resources. An attempt to access such a resource is intercepted via a filter invoked by the relevant ORB. The request is in turn evaluated by Akenti prior to allowing access to the actual resource server. The ORB need not be modified: Akenti is invoked via the OMG[8] standard interceptor mechanism. The server need only be modified to provide an interface module between the existing CORBA interceptor mechanism and the Akenti policy engine.

Akenti requires a communications protocol that provides, at minimum, for authentication of the communicating parties. For integration into the CORBA environment, Akenti utilizes Iona's[9] support for SSL in its Orbix ORB product.

Note that CORBA defines an extensive security service. However, authorization in the CORBA security model is based on access control lists (ACLs). Akenti provides greater flexibility than centralized ACLs by allowing distributed access control information in the form of digitally signed documents (certificates). The latter both define the access control policy and certify that a user has met the conditions of the policy. No central administrator need manage all certificates: they are instead administered by the (possibly distributed) stakeholders.

Secure mobile agents infrastructure

Akenti access control is also being incorporated into a prototype mobile agent infrastructure called the Anchor toolkit. This prototype is intended to explore the usefulness of secure mobile agents in a heterogeneous distributed computing environment. Mobile agents will be utilized to monitor remote program executions



Akenti protection of a CORBA resource

both in real time and to produce audit trails. The real-time monitoring can be used to provide robustness by restarting nonresponsive servers or informing human agents when intervention is needed. Agents can also be used to facilitate version consistency of software running on many hosts.

The objectives of the agent tool kit are: to deploy agents across hosts in a secure and authenticated manner; to provide tools on the host machines to run the agents in a restricted environment; and to monitor their execution for success or failure. The major security challenge is to provide secure and uncorrupted transmission of these agents from host to host and to provide secure communication between the agents.

These agents are being implemented in Java to achieve platform independence and to take advantage of Java's security features. The Java security manager allows fine grain access control of resources at runtime. We have extended this security manager to use the Akenti policy engine: Akenti makes access control decisions that define the scope of an agent's abilities.

The IAIK[10] Java implementation of SSL and The Java Cryptography Extensions are being used for secure communications.

Resource brokering

To protect their constituent resources (e.g., CPUs and storage devices), large-scale distributed systems at present must contend with and rely upon the local security and access control provided by the underlying operating systems. However, the scale and heterogeneity of distributed systems argues against extending these access control mechanisms, which are designed for smaller, centralized systems.

Instead, we are planning to integrate the distributed Akenti access control model into the Globus[11] infrastructure to allow secure use of network bandwidth. In the Globus environment, bandwidth will be divided according to the IETF's differentiated services (diff-serv) proposals[12]. diff-serv explicitly addresses only aggregate flows, however. A bandwidth brokering system will use the aggregate bandwidth allocations provided by diff-serv to schedule high-bandwidth, end-to-end data flows. The scheduling will require reservation of bandwidth prior to use, rather than satisfying bandwidth requests immediately. As in the secure Web server and camera controller projects, Akenti will validate (authenticate and authorize) all requests for bandwidth reservation according to the policy established by the resource's stakeholders, which in this case will include both the sending and receiving hosts for the data flow. Akenti will also validate the attempt to start using the reserved bandwidth as well. However, its role will end upon establishment of the end-to-end flow, so the actual data transfer will be unencumbered by the overhead of access control.

Although our initial thrust is in reserving bandwidth, this system is extensible to allocating compute cycles, storage space, and other computing resources securely, and we have plans to do so as part of the Clipper project[13].

This is a work in progress.

---

[1] See http://www-itg.lbl.gov/Diesel/.
[2] See http://tpm.amc.anl.gov/mmc/.
[3] See http://www.mcs.anl.gov/cif/.
[4] See http://www.apache.org/.
[5] See http://developer.netscape.com/docs/manuals/security/sslin/index.htm.
[6] See http://www-itg.lbl.gov/mbone/devserv/.
[7] See http://www.omg.org/news/begin.htm.
[8] See http://www.omg.org/.
[9] See http://www.iona.com/.
[10] See http://jcewww.iaik.tu-graz.ac.at/index.htm.
[11] See http://www.globus.org/.
[12] See http://www.ietf.org/html.charters/diffserv-charter.html and http://diffserv.lcs.mit.edu/.
[13] See http://www-itg.lbl.gov/Clipper/.